BouleDeCristal

Au cours de ce tutoriel, vous allez créer une application qui fonctionne comme une boule de cristal : l'utilisateur réfléchit à une question, comme « Est-ce que je vais gagner le match de foot ? » puis il secoue le téléphone pour connaître la réponse de la boule de cristal !

Pour commencer

***Se connecter sur* App Inventor**

- Consulter http://appinventor.mit.edu/
- Se connecter en utilisant le nom d'utilisateur et le mot de passe du compte Gmail.
- Cliquer sur l'onglet *My Projects*, et cliquer sur *New*.
- Créer un nouveau projet appelé « BouleDeCristal ».

Télécharger les éléments extérieurs

- Télécharger le fichier de l'image de la boule de cristal de http://bit.ly/crystalballTC

Programmer l'interface de l'application dans le *Designer*

Créer *Screen1*

REMARQUE: C'est une bonne idée de prendre l'habitude de créer un écran « Accueil » pour les applications, puisqu'on ne peut pas mettre d'écran avant Screen1 avec App Inventor. Donc si on programme *Screen1* en tant qu'écran principal, on ne peut plus ajouter d'écran accueil parce qu'on ne l'a pas fait depuis le début. Nous vous montrerons comment créer un écran d'accueil qui est relié à l'écran principal dans les tutoriels *BouleDeCristal* et *PotDePeinture*.

***Ajouter une étiquette Bienvenue à* Screen1**

- Tirer une *Label* de la *Basic Palette* jusqu'à l'écran.
- Cliquer sur *Rename* et la renommer « EtiquetteBienvenue ».
- Dans le panneau de configuration *Properties*, changer le champ de saisie *Text* à « Bienvenue ».

***Ajouter le bouton Entrer à* Screen1**

- Tirer un *Button* de la *Basic Palette* jusqu'à l'écran.
- Cliquer sur *Rename* et le renommer « BoutonEntrer ».
- Dans le panneau de configuration *Properties*, changer le champ de saisie *Text* à « Accéder à la Boule de Cristal ».

Créer l'écran BouleDeCristal

Ajouter l'écran BouleDeCristal

- Cliquer sur le bouton *Add Screen* du menu en haut de l'onglet *Design*.
- Sélectionner l'élément *Screen* dans le *Designer*.
- Dans le panneau de configuration *Properties* changer le champ de saisie *Title* à « BouleDeCristal ». Vous devriez maintenant voir un nouveau titre dans le champ de saisie.

Ajouter les étiquettes Mode d'emploi

- Tirer une *Label* de la *Basic Palette* jusqu'à l'écran.
- Cliquer sur *Rename* et la renommer « EtiquetteQuestion ».
- Dans le panneau de configuration *Properties*, changer le champ de saisie *Text* à « Que voudriez-vous savoir ? »

- Tirer une *Label* de la *Basic Palette* jusqu'à l'écran.
- Cliquer sur *Rename* et la renommer « EtiquetteModeEmploi ».
- Dans le panneau de configuration *Properties*, changer le champ de saisie *Text* à « Secouer la boule de cristal et tout vous sera dévoilé… »

Ajouter la boule de cristal

- Tirer un *Button* de la *Basic Palette* jusqu'à l'écran.
- Cliquer sur *Rename* et le renommer « TextePrediction ».
- Dans le panneau de configuration *Properties*, effacer « TextForButton1 » du champ de saisie *Text*.
- Dans le panneau de configuration *Properties*, attribuer à *BackgroundImage* l'image de la boule de cristal téléchargée ultérieurement. Cliquer sur *None* et choisir *Add* pour télécharger le fichier image.

***Ajouter* Accelerometer**

- Dans le panneau de configuration *Sensor*, tirer l'élément *Accelerometer* jusqu'à l'écran.

Programmer dans le *Blocks Editor*

Ecrire l'algorithme pour aller de* Screen1 *à l'écran BouleDeCristal

- Dans le *Designer*, sélectionner *Screen1* pour qu'il apparaisse dans le panneau de configuration *Viewer*.
- Tirer du *Blocks Editor*, sous *My Blocks*, un gestionnaire d'évènement *When…Clicked event handler* pour BoutonEntrer.
- Dans la colonne *Built-in*, ouvrir le tiroir *Control* et en tirer le bloc *call open another screen*. Placer ce bloc à l'intérieur du gestionnaire d'évènement *click*.
- Dans la colonne *Built-in*, ouvrir le tiroir *Text* et en tirer une boîte *Text* et l'attacher au bloc *call open another screen*. Changer le texte à « BouleDeCristal ».

Programmer l'écran BouleDeCristal

- Dans le *Designer* sélectionner l'écran *BouleDeCristal* pour qu'il soit visible dans le panneau *Viewer*.
- Retourner au *Blocks Editor*, il est maintenant vide.

Ecrire l'algorithme pour faire apparaître une réponse quand le téléphone est secoué

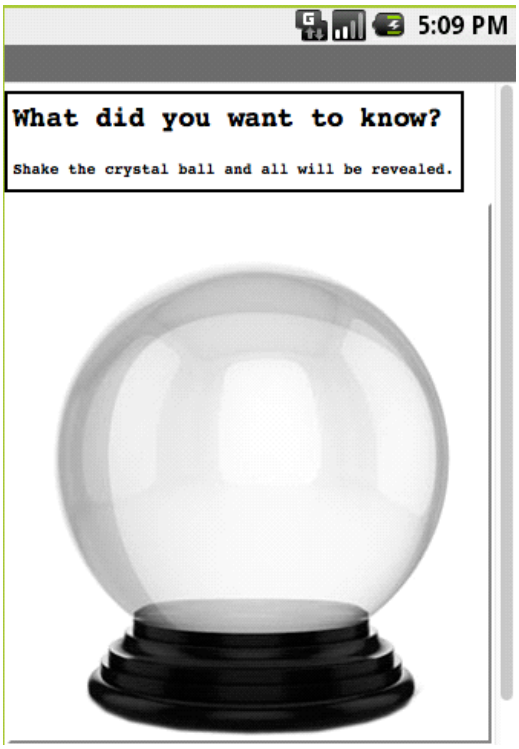
- Dans le *Blocks Editor* sous *My Blocks*, sélectionner le gestionnaire d'évènement *Shaking Event Handler* pour l'*Accelerometer*.
- Dans la colonne *My Blocks*, ouvrir le tiroir TextePrediction et en tirer *set TextePrediction.Text*. Placer ce bloc à l'intérieur du gestionnaire d'évènement *when … Clicked event handler.*
- Dans la colonne *Built-In*, ouvrir le tiroir *Lists*. En tirer le bloc *call pick random item.* Attacher ce block au bloc *set TextePrediction.Text.*
- Dans la colonne *Built-In*, ouvrir le tiroir *Lists*. En tirer le bloc *make a list* et l'attacher au bloc *call pick random item*.
- Dans la colonne *Built-In*, ouvrir le tiroir *Text*. En tirer 4 boîtes *Text* et les ajouter à la *list* en tant que *items.* Changer le texte à « Oui », « Non », « Peut-être » et « Qui vivra verra ».

Ecrire l'algorithme pour effacer la réponse quand on clique sur l'écran

- Tirer depuis le *Blocks Editor* dans *My Blocks* un gestionnaire d'évènement *when … Clicked event handler* pour le bouton TextePrediction.
- Dans la colonne *My Blocks*, ouvrir le tiroir TextePrediction et en tirer *set TextePrediction.Text*. Placer ce bloc à l'intérieur du gestionnaire d'évènement *when … Clicked event handler.*
- Dans la colonne *Built-In*, ouvrir le tiroir *Text*. En tirer une boîte *Text* et l'attacher au bloc *set TextePrediction.Text*. Effacer le texte.

Tester et télécharger l'application

- Sauvegarder l'application depuis le *Blocks Editor* et depuis le *Designer*.
- Connecter le téléphone grâce au câble USB, puis cliquer sur « *Connect to Device* » dans le coin en haut à droite du *Blocks Editor*. Sélectionner le téléphone de la liste et vérifier que le téléphone et la flèche jaunes clignotent et deviennent vertes. Vous saurez que le téléphone est connecté quand ils sont verts, et l'application devrait apparaître sur l'écran du téléphone.
- Tester l'application pour vérifier qu'il marche, puis la télécharger : dans le *Designer*, choisir « Package for Phone » et attendre qu'elle soit téléchargée, ce qui peut prendre quelques minutes.



Piratez votre application!

- Changer la police de caractère, les couleurs, etc.
- Ajouter de nouvelles réponses à la liste!
- Ajouter un son de « carillon » quand l'application est secouée.

Mettez vos questions et une capture d'écran de votre application « piratée » sur P2PU.

PotDePeinture

Au cours de ce tutoriel, vous allez créer une application qui permet à l'utilisateur de dessiner sur l'écran des traits et des points en couleur.

Pour commencer

- Se connecter sur App Inventor.
- Créer un nouveau projet appelé « PaintPot ».

Programmer l'interface de l'application dans le *Designer*

Créer *Screen1*

***Ajouter une étiquette Bienvenue à* Screen1**

- Tirer une *Label* de la *Basic Palette* jusqu'à l'écran.
- Cliquer sur *Rename* et la renommer « EtiquetteBienvenue ».
- Dans le panneau de configuration *Properties*, changer le champ de saisie *Text* à « Bienvenue ».

***Ajouter le bouton Entrer à* Screen1**

- Tirer un *Button* de la *Basic Palette* jusqu'à l'écran.
- Cliquer sur *Rename* et le renommer « BoutonEntrer ».
- Dans le panneau de configuration *Properties*, changer le champ de saisie *Text* à « Accéder à Pot de Peinture»

Créer l'écran PotDePeinture

Ajouter l'écran PotDePeinture

- Cliquer sur le bouton *Add Screen* du menu en haut de l'onglet *Design*.
- Sélectionner l'élément *Screen* dans le *Designer*.
- Dans le panneau de configuration *Properties* changer le champ de saisie *Title* à « PotDePeinture ». Vous devriez maintenant voir un nouveau titre dans le champ de saisie.

Ajouter les boutons couleurs

- Maintenant, vous allez ajouter 3 *buttons* pour changer la couleur de la peinture. Pour commencer, ajouter à l'écran un *button* appelé « BoutonRouge ». Changer la couleur de fond au Rouge dans le panneau de configuration *Properties* et le texte à « Rouge ».
- Créer un autre *button* appelé « BoutonVert » de couleur verte et de texte affiché « Vert ».
- Créer un troisième *button* appelé « BoutonBleu » de couleur bleue et de texte affiché « Bleu ».

Organiser les boutons

- Aligner les boutons côte à côte en les tirant dans un *Horizontal Arrangement*. Vous devez fixer la largeur de la disposition et des boutons à « *Fill parent... »* pour faire en sorte que les boutons prennent toute la place sur l'écran. *La composition est-elle améliorée ? Comment a-t-elle changée ?*

Créer un espace de dessin

- Ajouter une *canvas* appelée « ToileDessin ». Fixer la hauteur à 300 pixels et la largeur pour qu'elle prenne toute la largeur de l'écran (Indice : vous avez fait la même chose pour le *Horizontal Arrangement*).

Ajouter un bouton Effacer

- Ajouter un autre *button* appelé « BoutonEffacer » de texte affiché « Effacer ». Ce *button* effacera l'écran.

Programmer dans le *Blocks Editor*

- Ouvrir le *Blocks Editor* et le connecter au téléphone.

Ecrire l'algorithme pour aller de Screen1 à l'écran PotDePeinture

- Dans le *Designer,* sélectionner *Screen1* pour qu'il apparaisse dans le panneau de configuration *Viewer*.
- Dans le *Designer* sélectionner *Screen1* pour qu'il soit visible dans le panneau *Viewer*.
- Tirer depuis le *Blocks Editor* sous *My Blocks*, un gestionnaire d'évènement *When...Clicked event handler* pour *BoutonEntrer*.
- Dans la colonne *Built-in*, ouvrir le tiroir *Control* et en tirer *call open another screen*. Placer le bloc à l'intérieur du gestionnaire d'évènement *click.*
- Dans la colonne *Built-in*, ouvrir le tiroir *Text* et en tirer une boîte *Text.* L'attacher au bloc *call open another screen* et changer le texte à « PotDePeinture ».

Programmer l'écran PotDePeinture

- Dans le *Designer* sélectionner l'écran *PotDePeinture* pour qu'il soit visible dans le panneau *Viewer*.
- Retourner au *Blocks Editor*; il est maintenant vide.

Programmer les boutons couleurs

- Programmer les boutons couleurs. Premièrement, trouver le gestionnaire d'évènement *click event handler* pour BoutonRouge et le tirer jusqu'au *Blocks Editor.* Il affichera « when RedButton.Click do. » Maintenant vous allez le remplir avec des blocs pour lui dire ce qu'il faut faire quand on clique sur le BoutonRouge. Trouver le bloc qui choisit la couleur de la toile et le mettre dans le gestionnaire d'évènement BoutonRouge. Il affichera « set DrawingCanvas.PaintColor to. » Lui attacher un bloc couleur rouge (indice : regarder dans les blocs « Built-in »). *Qu'accomplit cet algorithme ?*
- Maintenant programmer les boutons vert et bleu de la même manière.

Programmer le bouton Effacer

- Programmer le bouton Effacer. Trouver le gestionnaire d'évènement *click* de BoutonEffacer (comme pour les boutons couleurs). Maintenant le programmer pour qu'il efface la toile. (Indice : il y a un bloc pour la ToileDessin qui a pour fonction *ToileDessin.clear*).

Ecrire l'algorithme pour dessiner les points

- Maintenant vous allez ajouter un algorithme pour qu'un point soit dessiné à l'endroit où la toile est touchée. Trouver le gestionnaire d'évènement *touched* de ToileDessin et le tirer jusqu'au Blocks Editor. Les variables *x* et *y* fournis par le gestionnaire d'évènement représentent les coordonnées du point où l'écran est touché. Utiliser la DrawCircle Procedure avec les valeurs de *x* et *y* fournis sous « My Definitions » pour dessiner un cercle au point où l'écran est touché. La variable *r* de la procédure DrawCircle représente le rayon du cercle en pixels. 5 est une bonne valeur pour commencer. (Indice : regarder sous les blocs « Built-in »).

Tester l'application

- Tester les boutons couleur, le bouton Effacer, et ce qui se passe quand on touche l'écran du téléphone. *Les boutons changent-ils la couleur de la peinture ? Le cercle est-il dessiné là où vous avez touché ? Le bouton Effacer efface-t-il toute la peinture ?*

Ajouter l'algorithme pour dessiner les lignes

- Maintenant vous allez ajouter l'algorithme pour que quand l'utilisateur dessine avec son doigt (en tirant) sur la toile, l'application trace une ligne. Trouver le gestionnaire d'évènement *dragged* de *ToileDessin*. Un évènement *drag* vient avec 7 arguments. Il y a trois couples de coordonnées x et y qui montrent la position de votre doigt où la trace a commencé, la position actuelle du doigt, et la position instantanément ultérieure du doigt. Il y a aussi *sprite*, que nous n'utiliserons pas pour ce tutoriel. Maintenant trouver la procédure ToileDessin appelée *DrawLine*. Cette procédure dessine un trait de la position précédente (x1, y1) à la position actuelle (x2, y2). Modifier les arguments pour qu'elle dessine un trait depuis la position précédente du doigt à la position actuelle du doigt.

Tester votre application

- Tester votre application. Essayer de dessiner des droites et des courbes.

Ajouter des variables pour dessiner des points de tailles différentes

- Maintenant vous allez améliorer votre application en ajoutant deux boutons pour changer la taille des points dessinés quand l'écran est touché. Premièrement ajoutons une variable appelée TaillePoint. Fixer la valeur initiale de TaillePoint à 8. Modifier la procédure DrawCircle pour qu'elle dessine un point de rayon 8 (se rappeler qu'avant vous l'aviez fixé à 5 pixels). Maintenant quand l'écran est touché, le cercle dessiné sera de la taille du nombre affecté à la variable TaillePoint.
- Maintenant dans le *Designer*, ajouter un BoutonGros et un BoutonPetit à l'écran. Utiliser un autre *horizontal arrangement* pour les aligner avec le bouton Effacer. Faire en sorte que les boutons affichent le texte « Gros Points » et « Petits Points ».
- Dans le *Blocks Editor*, trouver le gestionnaire d'évènement *click* de BoutonPetit et utiliser-le pour affecter 2 à la variable TaillePoint quand le BoutonPetit est cliqué. Maintenant faisons de même pour GrosBouton, mais affecter 8 à TaillePoint.

Tester votre application

- Tester votre application sur le téléphone.

« Piratez » votre application!

- Personnaliser votre application : utiliser l'élément appareil-photo pour prendre des photos de vos amis et de les mettre comme fond d'écran pour que vous puissiez dessiner dessus.

Ressources

Vidéos YouTube

- Vidéo du Professor Wolber, Partie 1: http://www.youtube.com/watch?v=gHPae2B5pN0
- Vidéo du Professor Wolber, Partie 2: http://www.youtube.com/watch?v=L1_ATVJiiLI
- Vidéo du Professor Wolber, Partie 3: http://www.youtube.com/watch?v=XGXFJxrYBc0
- Vidéo Caméra, Professor Wolber: http://www.youtube.com/watch?v=q39n6JwrdPk
- Vidéo de Chris Groff, Partie 1: http://www.youtube.com/watch?v=31bi9tWOLcY
- Vidéo de Chris Groff, Partie 2: http://www.youtube.com/watch?v=JX8Gn7FLcHY

Mettez vos questions et une capture d'écran de « piratée » sur P2PU.

JeuDeLaTaupe

Au cours de ce tutoriel, vous allez créer un jeu : une taupe « saute » tous les 0,5 secondes, partout sur l'écran d'une manière aléatoire, et si l'utilisateur réussit à toucher la taupe, le score augmente d'un point.

Pour commencer

- Se connecter sur App Inventor
- Créer un nouveau projet appelé « JeuDeLaTaupe ».
- Télécharger l'image de la taupe de bit.ly/moleimage

Programmer l'interface de l'application dans le *Designer*

Changer le titre de l'écran

- Changer le titre de l'écran à « JeuDeLaTaupe » depuis le panneau de configuration *properties* dans le *Designer*. *Pouvez-vous voir le nouveau titre affiché sur le téléphone ?*

Ajouter la toile, les boutons et le son

- Ajouter une *canvas* appelée « MaToile ». Fixer la hauteur à 300 pixels et la largeur à la largeur de l'écran. (Indice : c'est pareil que dans PotDePeinture).
- Maintenant, vous allez ajouter un *button* pour redémarrer le jeu. Ajouter un bouton sur l'écran appelé « BoutonRedémarrer ». Changer le texte affiché à « Redémarrer ».
- Maintenant ajouter une *label.* Vous allez utiliser cette étiquette pour afficher le score, donc appelons-la « EtiquetteScore ».Changer le texte à « Score : — ».
- Ensuite, ajouter l'élément *sound.* Remarquer que quand vous le tirer jusqu'à la toile il apparaît en bas de l'écran. *Pourquoi pensez-vous qu'App Inventor fait cela ?* C'est parce que le son est un composant non-visible. Renommer l'élément son « Bruit ».

***Ajouter un élément* Clock**

- Maintenant, vous allez ajouter un élément *clock* qui servira de chronomètre, pour que l'application sache quand il faut bouger la taupe. Renommer la *clock* « ChronoTaupe », et fixer son *TimeInterval* à « 500 » millisecondes pour faire bouger la taupe toutes les 0,5 secondes.
- Vérifier que *TimerEnabled* est coché, pour que le chronomètre soit actif.

Ajouter la taupe

- En utilisant l'*Animation Palette,* ajouter une *ImageSprite,* et l'appeler « Taupe ». Mettre comme image, l'image de la taupe téléchargée précédemment. Vérifier que l'*ImageSprite* est activé, et que *Visible* est coché. *Height* et *Width* devraient être fixé à leur taille automatique.
- Maintenant, vous aller modifier les propriétés qui contrôlent le mouvement de la taupe. La position initiale (représentée par les coordonnées X et Y) est déterminée par où vous avez placé la taupe quand vous l'avez tiré jusqu'à MaToile. Tirer la taupe jusqu'à une position différente. Vous devriez voir X et Y changer.
- Le *Heading* représente la direction dans laquelle la taupe va bouger. Fixons-le à 0. Ensuite fixer *Speed* à 0 puisqu'on ne veut pas que la taupe bouge en une ligne. Vous allez plutôt la faire « sauter » partout en changeant la position. Fixer l'intervalle à 500 puisqu'on veut que la taupe bouge toutes les 500 millisecondes.

Programmer dans le *Blocks Editor*

- Ouvrir le *Blocks Editor* et connecter le téléphone.

Ecrire une procédure qui contrôle le mouvement de la taupe

- Ajouter un bloc *procedure* du tiroir *definition.* Vous pouvez cliquer sur le texte « procedure » pour renommer le bloc. Le renommer « BougerTaupe ».
- Dans la partie *do* de la procédure BougerTaupe, ajouter le bloc qui bougerait la taupe à une nouvelle localisation x et y. (Indice : Il sera sous la section Taupe.) Ensuite changer les coordonnées *x* et *y* pour contrôler là où la taupe va bouger. Les coordonnées peuvent être un nombre quelconque du moment que c'est sur la toile. La coordonnée *x* peut être entre 0 (le côté gauche de la toile) et la largeur de la toile moins la largeur de la taupe. La coordonnée *y* peut être entre 0 (le bas de la toile) et la hauteur de la toile moins la hauteur de la taupe. Vous voulez que la taupe bouge partout à une localisation imprévisible, donc vous allez définir les valeurs de *x* et *y* comme des nombres aléatoires dans les intervalles décrits précédemment. Vous trouverez le bloc *random* dans le tiroir *Math*. Fixer le nombre *from* à 0. Dans l'emplacement *to*, effacer le bloc *number* et ajouter à sa place un bloc *subtraction*. Affecter la valeur *MaToile.Width* moins *Taupe.Width* à la coordonnée *x* à et la valeur de *MaToile.Width* moins *Taupe.Width* à la coordonnée *y*.
- Maintenant (soit avec le téléphone ou le programme d'émulation connecté au Blocks Editor), tester la procédure Bouger.Taupe en faisant un clic-droit et choisissant « Do It ». *Que se passe-t-il ? Est-ce que c'est ce à quoi vous vous attendiez ?*

Ecrire l'algorithme pour activer la procédure Bouger.Taupe toutes les 0,5 secondes

- Prendre le bloc ChronoTaupe.Timer et le faire activer (call) la procédure *Bouger.Taupe* dans *My Definitions. Que fait cet algorithme ? Comment sait-il de le faire toutes les 0,5 secondes ? Indice : comment avez-vous fixé les propriétés ?*

Tester votre application

- Tester l'application. *La taupe bouge-t-elle aléatoirement toutes les 0,5 secondes ?*

Créer une procédure pour changer le score

- Créer une *variable* appelé « score » pour représenter le score. Fixer la valeur initiale à 0.
- Prendre un bloc de *procédure* et l'appeler « MiseAJourScore ». A l'intérieur mettre le bloc qui change le texte d'EtiquetteScore.
- Vous allez changer le texte à « Score : » suivi de la valeur du score, donc il faut utiliser un *join.* Le bloc *join* se trouve dans le tiroir *Text*.
- Modifier la première moitié du *join* au texte « Score : » et la deuxième à la *variable* que vous avez créé pour le score.

Ajouter l'algorithme qui augmente le score quand la taupe est touchée

- Trouver le gestionnaire d'évènement pour quand la Taupe est touchée. *Vous souvenez-vous des gestionnaires d'évènements de PotDePeinture ?*
- Premièrement vous allez utiliser le bloc *set global* pour changer la variable Score. Il vous faut un nouveau bloc *addition* pour fixer la nouvelle valeur du score à la valeur précédente + 1.
- Ensuite, on veut afficher le nouveau score, donc il faut activer (call) la procédure *MiseAJourScore* que vous avez créé pour changer l'étiquette, dans *My Definitions.*
- Activer (call) *Son.Vibrate* pendant 100 millisecondes pour que le téléphone vibre, depuis *My Blocks > Noise* et activer BougerTaupe pour que la taupe bouge.

Tester l'application

- Tester l'application en jouant le jeu. *Le score augmente-il quand la taupe est touchée ?*

Programmer le bouton Redémarrer pour remettre le score à 0

- Trouver le gestionnaire d'évènement *click* pour le bouton Redémarrer. Prendre le bloc *set global* score et le placer à l'intérieur. Fixer la valeur du score à 0. Encore une fois, il faut activer (call) *MiseAJourScore*, pour que le nouveau score apparaisse sur l'écran.

Tester votre application

- Tester l'application en jouant le jeu et en utilisant le bouton Redémarrer. *La taupe bouge-t-elle aléatoirement toutes les 0,5 secondes ? Le score augmente-il quand la taupe est touchée ? Le bouton Redémarrer fonctionne-t-il ?*

« Piratez » votre application !

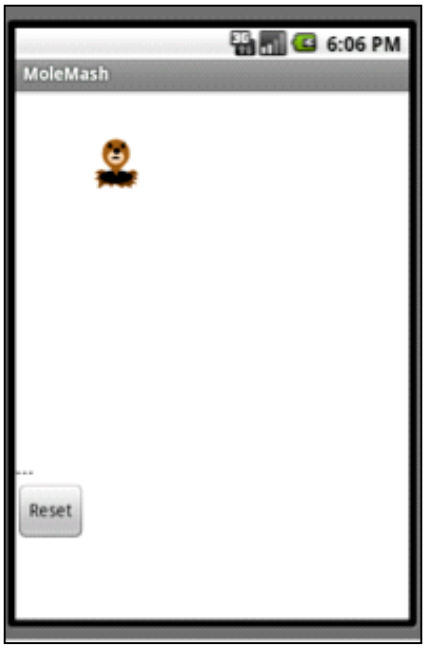
- Ajouter un *son* au jeu, pour que le téléphone joue un son au lieu de vibrer. Vous devez mettre à jour l'élément *sound* et le gestionnaire d'évènement *touched* de la taupe.
- Et si on ajouter une deuxième taupe ?

Ressources

Vidéos YouTube

- Vidéo de Chris Groff Part 1: http://www.youtube.com/watch?v=X-Ozwa-BqG8
- Vidéo de Chris Groff Part 2: http://www.youtube.com/watch?v=BOprKoivALQ
- Vidéo de Chris Groff Part 3: http://www.youtube.com/watch?v=iNzLmqNtW5c
- Vidéo du Professeur Wolber Partie 1: http://www.youtube.com/watch?v=uj8b19S1F50
- Vidéo du Professeur Wolber Partie 2: http://www.youtube.com/watch?v=sP30C29J44w
- Vidéo du Professeur Wolber Partie 3: http://www.youtube.com/watch?v=NCgmyoHWiE4
- Vidéo du Professeur Wolber Partie 4: http://www.youtube.com/watch?v=3BW0E9Gulhk
- Vidéo du Professeur Wolber Partie 5: http://www.youtube.com/watch?v=tkjjDf0a3ZA

Mettez vos questions et une capture d'écran de votre application « piratée » sur P2PU



Inventer Votre Propre Application

Travailler en autonomie ou en binôme pour créer et développer votre propre application.

Essayer d'utiliser les outils que vous avez appris :

- Ecran Accueil
- Evènements
- Variables
- Listes
- Procédures
- Appareil Photo
- Boutons
- Accéléromètre
- Etiquette
- Image *Sprites*
- Chronomètres
- Sons

Obtenir les impressions de l'utilisateur

- Montrer l'application à votre groupe, votre famille, vos amis. *Est-ce qu'ils ont aimé ? Est-ce qu'ils utiliseraient cette application ? Est-ce que c'est facile à comprendre/utiliser ? Est-ce qu'ils ont des idées pour l'améliorer ?*

Ressources

Vidéos sur YouTube

- Regarder les vidéos ou les indications des « piratages » précédents si vous voulez vous rappeler de comment utiliser un certain élément.

Mettez vos questions et une capture d'écran de votre application « piratée » sur P2PU.

Pas de SMS au Volant

Au cours de ce tutoriel, vous allez créer une application qui répond automatiquement aux SMS pendant que l'utilisateur est en train de conduire. L'utilisateur a aussi l'option de faire « lire » à haute voix le message qui a été reçu.

Pour commencer

- Se connecter sur App Inventor
- Créer un nouveau projet appelé " PasDeSMSAuVolant "

Programmer l'interface de l'application dans le *Designer*

Changer le titre de l'écran

- Changer le titre de l'écran à « Pas de SMS au Volant » dans le panneau de configuration *Properties* du *Designer. Pouvez-vous voir le nouveau titre affiché sur le téléphone ?*

Créer l'interface

- Ajouter une *Label* appelée « EtiquetteEntrerTexte » affichant le texte « Le texte ci-dessous sera envoyé en réponse à tout message SMS reçu. »
- Ajouter une *Label* appelée « EtiquetteReponse » affichant le texte « Je suis en train de conduire à l'instant, je vous rappellerai prochainement. »
- Changer la police, les couleurs, etc. pour que l'écran soit comme vous le souhaitez.
- Ajouter une *Textbox* appelée « ZoneDeTexteNouvelleReponse » et affecter « » au texte. *Que va-t-il se passer quand on fixe le texte à « » ?* Changer le *Hint* à « Taper le nouveau message SMS. »
- Ajouter un *Button* appelé « BoutonEntrerReponse » affichant le texte « Modifier la réponse ».

Ajouter les éléments non-visibles

- Ajouter les éléments : *Texting*, *TinyDB,* et *TextToSpeech*.

Programmer dans le *Blocks Editor*

- Ouvrir le Blocks Editor et connecter le téléphone.

Programmer une réponse automatisée

- Vous allez écrire un algorithme pour que l'application envoie une réponse automatisée quand quelqu'un envoie un SMS au téléphone.
- En utilisant le composant *Texting*, trouver le gestionnaire d'évènement *Message Received* et la fonction *Send Message*. Placer les blocs pour que l'algorithme envoie un message quand un message est reçu.
- Vous devez modifier les propriétés *PhoneNumber* et *Message* de l'élément *Texting* avant d'activer (call) la fonction *Send Message*. Indice : utiliser le numéro de *Message Received* pour définir le numéro de téléphone à appeler (c'est généré automatiquement dans le tiroir *My Definitions*.) Le *Message* doit être changé au texte d'EtiquetteReponse.

Tester l'application

- Tester l'application en envoyant un SMS au téléphone. *Le téléphone envoie-t-il la réponse automatisée ?* Vous pouvez le tester dans le programme d'émulation en utilisant l'identifiant de l'émulateur (par ex : 5554) comme numéro de téléphone.

Entrer une réponse personnalisée

- Maintenant vous allez écrire l'algorithme qui permet à l'utilisateur de changer la réponse qu'envoie son téléphone. Trouver le gestionnaire d'évènement *click* de BoutonEntrerReponse. Utiliser le bloc *Text* de *Set EtiquetteReponse* pour changer le texte à *ZoneDeTexteNouvelleReponse.Text* quand le bouton est cliqué.

Tester votre application

- Tester l'application en entrant une nouvelle réponse et en envoyant un SMS au téléphone. *Le téléphone qui a envoyé le message reçoit-il la nouvelle réponse automatisée ?*

Stocker les réponses personnalisées dans une base de données

- *Que se passe-t-il à la réponse personnalisée quand on quitte puis on rouvre l'application ?* Pour stocker les réponses personnalisées de l'utilisateur, il faut une *base de donnée*s. Le composant *TinyDB* nous permet de stocker des données (*StoreValue*) et de récupérer des données (*GetValue*).
- Modifier le gestionnaire d'évènement *click* de BoutonEntrerReponse que vous avez déjà programmé pour stocker la réponse. Affecter le texte « MessageReponse » au champ *tag*. Vous utiliserez ce *tag* pour récupérer le message plus tard. Affecter *valueToStore* à *EtiquetteReponse.Text*. Vérifier que vous aviez mis un nouveau message dans EtiquetteReponse avant de le stocker.

Récupérer les réponses personnalisées quand l'application est initialisée

- App Inventor fournit un gestionnaire d'évènement spécifique (*Screen1.Initialize*) qui commence quand l'écran est ouvert, c'est-à-dire quand l'application est initialisée. Vous pouvez utiliser *Screen1.Initialize* pour affecter la réponse personnalisée stockée dans la base de données à EtiquetteReponse quand l'application est initialisée. Cependant, il se peut qu'aucune réponse personnalisée n'ait été stockée, donc il faut vérifier la réponse récupérée de la base de données avant de l'affecter à EtiquetteReponse.
- Créer une *variable* appelé « reponse » et lui affecter un texte vide.
- Dans le gestionnaire d'évènement *initialize*, affecter la valeur récupérée de *CallTinyDB1.GetValue* à la *variable «* reponse ». Souvenez-vous qu'il faut affecter « MessageReponse » au *tag* pour récupérer une réponse de la base de données.
- En utilisant un bloc *If then*, mettre comme condition : la longueur de la *variable* « reponse » est plus grande que 0. Indice : utiliser le bloc *Call Length text* et le bloc supérieur à (>).
- Changer la partie *then-do* du bloc pour que l'application affecte la *variable* « reponse » à ResponseLabel.Text.

Tester votre application

- Regarder ce qui se passe quand vous ouvrez et fermez le logiciel sans sauvegarder de message personnalisé. *Quel message automatisé est affiché ?*
- Regarder ce qui se passe quand vous ouvrez et fermez le logiciel après avoir sauvegarder un message personnalisé. *Ce message personnalisé apparaît-il ?*

Lire les nouveaux messages à voix haute

- Maintenant vous allez modifier l'application pour que quand l'utilisateur reçoit un SMS, le numéro de téléphone et le message reçu soient lus à voix haute. Ajouter le bloc *TextToSpeech1.Speak* à la fin du gestionnaire d'évènement *Message Received.*
- Pour identifier le message qui va être lu, utiliser un bloc *Call Make Text*. Changer la première « valeur » (= value) du texte à « SMS reçu de ». N'oublier pas l'espace après « de ». Changer la deuxième « valeur » du texte au numéro de téléphone qu'on trouve dans le gestionnaire d'évènement *Message Received.* Changer la troisième valeur du texte à « Le message est ». N'oublier pas l'espace après « est ». Changer la dernière valeur du texte à *messageText*, qu'on trouve dans le gestionnaire d'évènement *Message Received.* On vient de créer un seul message qui « lit » le numéro de téléphone est le message à l'utilisateur.

Tester l'application

- Tester l'application en envoyant un SMS au téléphone. *Le téléphone « lit »-il le numéro et le message à haute voix ?* Se rappeler de tester toutes les caractéristiques techniques de l'application pour s'assurer qu'elles marchent toujours.

« Piratez » votre application!

- Développer une version qui permet à l'utilisateur de définir des réponses personnalisées pour les différents numéros de téléphones.
- Défi : Utiliser l'élément *LocationSensor* pour envoyer la localisation actuelle du conducteur.

Ressources

Vidéos sur YouTube

- Vidéo de Professor Wolber, Partie 1: http://www.youtube.com/watch?v=rrwrfX8T_c8
- Vidéo de Professor Wolber, Partie 2: http://www.youtube.com/watch?v=0FTdv1SxGmc
- Vidéo de Professor Wolber, Partie 3: http://www.youtube.com/watch?v=7PmDsrGHIZI
- Vidéo de Professor Wolber, Partie 4: http://www.youtube.com/watch?v=5RIdqkJgfn4
- Vidéo de Professor Wolber, Partie 5: http://www.youtube.com/watch?v=R9MPcbe_oo4

Mettez vos questions et une capture d'écran de votre application « piratée » sur P2PU.

12:16 PM
No Texting While Driving
The text below will be sent in response to all incoming SMS messages while this app is running:
I'm driving right now, I'll contact you shortly.
enter new response text
Modify Response